REMARKS

The Examiner has rejected Claims 1-30 under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. Specifically, the Examiner has argued that 'the limitation "tangible computer readable medium" is considered new matter' since "[t]he [E]xaminer did not find support for this limitation in the original disclosure." Applicant respectfully disagrees, and asserts that applicant's claimed "tangible computer readable medium" is supported in the original disclosure. For example, see applicant's disclosure of "cache memory" in the Specification on Page 9, lines 26-27, et al.

Furthermore, the Examiner has rejected Claims 1-30 under 35 U.S.C. 101, as being directed towards non-statutory subject matter. Specifically, the Examiner has argued that applicant's claims "are directed to an algorithm, which is an abstract idea that [does] not correspond to any specific real world data," and that therefore the claims 'do not claim any "practical application" or "useful, concrete and tangible result".' Applicant respectfully disagrees and asserts that applicant's claimed "receiving instructions from the video memory" (see this or similar, but not necessarily identical language in independent Claims 1, 24, 25, 26, 28, 29, and 30) and "[retrieving] an instruction object stored in the frame buffer" (see independent Claim 27) provide a practical application, namely receiving instructions and retrieving instructions, as noted above, and also produce a tangible, useful, and concrete result. In addition, applicant respectfully points out that Claim 29 claims "outputting the processed pixels," which also clearly provides a tangible, useful, and concrete result.

The Examiner has rejected Claims 1-12, 18-21, 24-28, and 39 under 35 U.S.C. 103(a) as being unpatentable over Rivard (U.S. Patent No. 5,987,567), in view of Wang (U.S. Patent No. 5,831,640). Further, the Examiner has rejected Claims 13-17, 22, 23, and 29 under 35 U.S.C. 103(a) as being unpatentable over Rivard and Wang, and further in view of Applicant Admitted Prior Art (AAPA). Applicant respectfully disagrees with such rejections.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. *In re Vaeck,*947 F.2d 488, 20 USPQ2d 1438 (Fed.Cir.1991).

With respect to the first element of the *prima facie* case of obviousness, the Examiner has argued that "[o]ne of ordinary skill in [the] art... would have expected applicant's invention to perform equally well with Rivard's reference that teaches to send and receive information/instruction from the texture mapping stage and texel cache system to the DRAM because using [these] components together will also result in sending and receiving instructions to and from DRAM." To the contrary, applicant respectfully asserts that it would not have been obvious to modify the teaching of Rivard, especially in view of the vast evidence to the contrary.

Applicant respectfully points out that Rivard teaches that "[b]ecause memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030" (Col. 6, lines 62-66 - emphasis added). In addition, Rivard teaches that "[o]nce memory requests are generated, there is, depending on the DRAM design, a constant latency of about five to ten clock cycles (or possibly more) which includes time for exiting and reentering the graphics pipeline hardware, to effect a page hit," and "[t]herefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 6, line 66-Col. 7, line 7 - emphasis added). Furthermore, Rivard discloses that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 - emphasis added).

Applicant respectfully asserts that a combination of data and instructions in Rivard would result in no need for Rivard's purposefully included "pipeline latency elements," since the combination of data and instructions would arrive together. As a result, there is no suggestion that the modification of the Rivard reference, as argued by the Examiner, is desired. The mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination. *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990). Although a prior art device "may be capable of being modified to run the way the apparatus is claimed, there must be a suggestion or motivation in the reference to do so." 916 F.2d at 682, 16 USPQ2d at 1432.).

Additionally, applicant respectfully notes that modifying the Rivard reference, as suggested by the Examiner, would change the principle of operation of the Rivard system, since it would obviate the need for the purposefully included "pipeline latency elements 1025," as noted above. If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious. *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959)

Furthermore, applicant respectfully asserts that, in the Abstract, Rivard teaches "[a] system for caching texel information in a cache data store, for use in a graphics rendering system which uses interpolative sampling to compute texture color values." Additionally, Rivard teaches that "[t]he system includes a texel memory storing texel information, a graphics application program for using interpolative sampling to compute dynamic texture values, a first cache data storage for a number of the most-recently-retrieved texels, a second cache data storage for a previously-retrieved adjacent line of texels, cache tag blocks for determining whether the texels needed by the graphics accelerator system are cached in either of the first or second cache data stores, and a memory request generator for retrieving texels from texel memory upon indication of a miss by the cache tag blocks."

However, Rivard does not recognize one of the various possible problems solved by applicant, namely to "accommodate the programmability of recent texture and shader modules without being inhibited by the size of associated programs," for example. It was this insight in solving this problem that helped the inventors conceive of the claimed invention which overcomes the drawbacks of the prior art. "Because that insight was contrary to the understandings and expectations of the art, the structure effectuating it would not have been obvious to those skilled in the art." *Schenck v. Nortron Corp.*, 713 F.2d at 785, 218 USPQ at 700 (citations omitted).

Additionally, applicant respectfully asserts that the following excerpts from Rivard demonstrate that such reference, in fact, *teaches away* from applicant's claimed invention.

> "If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a <u>16-bit per pixel texture lookup</u> and the data is conveniently aligned in the <u>texture map</u>, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed." (Col. 6, lines 56-61 - emphasis added)
>
> "Therefore, graphics accelerator system 635 includes <u>pipeline latency</u> elements 1025 to coordinate arrival of the <u>memory data</u> and of the <u>associated instructions</u> at cache data store and memory data resolver 1030." (Col. 7, lines 3-7 - emphasis added)
>
> "…the memory request generator is coupled between the cache tag block and the cache data store for performing a <u>memory request before an address and instruction information associated with the needed texels</u> reach the cache data store." (Col. 10, lines 48-52 - emphasis added)

Applicant respectfully asserts that the "<u>pipeline latency elements 1025</u>," as described in Rivard, are introduced specifically "to <u>coordinate arrival of the memory data and of the associated instructions</u> at cache data store and memory data resolver 1030" (emphasis added). Thus, Rivard actually *teaches away* from applicant's claim language by intentionally incorporating the "pipeline latency elements 1025" for the specific purpose of coordinating the arrival of the instructions and the memory data from separate sources. A *prima facie* case of obviousness may also be rebutted by showing that the art, in any material respect, *teaches away* from the claimed invention. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed. Cir. 1997).

Thus, clearly at least the first element of the *prima facie* case of obviousness has not been met by the Rivard reference.

More importantly, with respect to the third element of the *prima facie* case of obviousness, applicant respectfully asserts that the Rivard reference also fails to meet all of applicant's claim limitations. Specifically, with respect to independent Claims 1, 24-27, and 30, the Examiner has relied on Figure 6 and Figure 10, in addition to the following excerpts from Rivard to make a prior art showing of applicant's claimed "sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory" (see this or similar, but not necessarily identical language in the foregoing independent claims).

> "Graphics application program 670 transfers graphical information from data storage 625 into DRAM 655 for local storage. Graphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650. Texture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645." (Col. 4, lines 46-57)

> "Cache data store 1030 responsively outputs on lines 649 the texture values cached in the read location.
>
> If a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020. Memory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval. DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address. If the interface to DRAM 655 is 32-

bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed.

Because memory request generator 1020 is between cache tag blocks 1010, 1015 and cache data store 1030, generator 1020 can perform DRAM 655 memory requests before the address and instruction information reach cache data store and memory data resolver 1030. Once memory requests are generated, there is, depending on the DRAM design, a constant latency of about five to ten clock cycles (or possibly more) which includes time for exiting and reentering the graphics pipeline hardware, to effect a page hit. Therefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030." (Col. 6, line 45-Col. 7, line 10)

Applicant respectfully asserts that the figures relied on by the Examiner only generally illustrate a computer system and a block diagram detailing the graphics accelerator system showing the pipeline latency elements 1025. In addition, the excerpts relied on by the Examiner merely teach that the "[m]emory request generator 1020 generates memory requests for all misses (up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests to DRAM 655 for information retrieval" (emphasis added). However, disclosing that a memory request is generated for all misses, as in Rivard, fails to teach "sending an instruction request to memory utilizing a texture module in a graphics pipeline" (emphasis added), as claimed by applicant.

Applicant notes that the Examiner has argued that "the texture mapping stage and the texel cache system together are considered the texture module." Applicant respectfully disagrees and asserts that Rivard merely suggests that the "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage **is separate from** the texture cache system, as clearly disclosed in

Rivard, and thus fails to suggest "sending an instruction request to memory utilizing a texture module in a graphics pipeline" (emphasis added), as claimed by applicant.

In addition, the Examiner has argued that "memory requests/read requests for all misses that are forwarded to DRAM are instruction requests based on which DRAM sends back information… [where] the DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Applicant respectfully disagrees, and asserts that Rivard merely discloses that the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57). Additionally, Rivard discloses that "[t]exel sample address computation block 1005 forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added).

However, Rivard's disclosure that the memory request generator 1020, which is included in the texel cache system (see Figure 10), generates memory requests for all misses and forwards the requests to DRAM for information retrieval, simply fails to even suggest "sending an instruction request to memory utilizing a texture module in a graphics pipeline" (emphasis added), as claimed by applicant. Moreover, the texel sample address computation block (included in texture mapping block 645) which forwards sample points to the cache tag blocks 1010 and 1015 (included in texel cache system 650), as disclosed in Rivard, fails to meet "sending an instruction request to memory utilizing a texture module in a graphics pipeline" (emphasis added), as claimed by applicant. Furthermore, the memory request to the DRAM, as in Rivard, fails to suggest "an instruction request," in the manner as claimed by applicant.

In addition, applicant respectfully asserts that the following excerpts from Rivard further demonstrate that the Rivard fails to disclose applicant's claimed "sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory" (see this or similar, but not necessarily identical language in the foregoing independent claims).

> "If the interface to DRAM 655 is 32-bits wide, the texture mode indicates a 16-bit per pixel texture lookup and the data is conveniently aligned in the texture map, then it is possible to satisfy two lookup requests with a single read request. However, if the data is not aligned, then two read requests are needed." (Col. 6, lines 56-61 - emphasis added)
>
> "Therefore, graphics accelerator system 635 includes pipeline latency elements 1025 to coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030." (Col. 7, lines 3-7 - emphasis added)
>
> "...the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store." (Col. 10, lines 48-52 - emphasis added)

First, applicant respectfully points out that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "memory data", where "the data is conveniently aligned in the texture map" (emphasis added), as in Rivard (see excerpts above), fails to teach "sending an instruction request" (emphasis added), as claimed by applicant.

Second, applicant respectfully asserts that Rivard simply discloses that a memory request is performed before an address and instruction information associated with the needed texels reach the cache data store (see excerpts above). Thus, the memory data in Rivard simply relates to texel data, which clearly fails to suggest "sending an instruction request to memory utilizing a texture module in a graphics pipeline" (emphasis added), as claimed by applicant.

Furthermore, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Applicant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Applicant respectfully asserts that Rivard merely discloses that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses… and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to suggest "sending an instruction request to memory utilizing a texture module in a graphics pipeline" (emphasis added), as claimed by applicant. Clearly, generating and forwarding a memory request to DRAM, as in Rivard, fails to suggest "sending an instruction request," particularly where "instructions [are received]…in response to the instruction request" (emphasis added), in the context as claimed by applicant.

Additionally, with respect to independent Claims 1, 24-27, and 30, the Examiner has relied on Figure 6 and Figure 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard (reproduced above) to make a prior art showing of applicant's claimed "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (see this or similar, but not necessarily identical language in the foregoing independent claims).

Specifically, the Examiner has argued that "DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage; [where] the texture mapping stage and the texel cache system together are considered the cache module, so the information/memory data is passed between the components of the texture module."

Applicant respectfully disagrees, and points out that in the second paragraph on Page 7 of the Office Action mailed 03/20/2007, the Examiner has stated that "Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module," and thus, Rivard simply fails to suggest "receiving instructions from the video memory in response to the instruction request <u>utilizing the texture module in the graphics pipeline</u>" (emphasis added), as claimed by applicant.

In addition, the excerpts from Rivard relied upon by the Examiner merely teach that "DRAM 655 <u>returns the **memory data** on bus 660 to cache data store</u> and memory data resolver 1030" (emphasis added). However, merely returning <u>memory data</u> to a cache data store, as in Rivard, simply fails to teach "receiving <u>instructions</u> from the video memory <u>in response to the instruction request</u> utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by applicant.

Furthermore, applicant respectfully asserts that Rivard teaches that "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 54-57) Additionally, Rivard discloses that "<u>[t]exel sample address computation block 1005</u> forwards the higher resolution sample points A-D to cache tag block 1010 and forwards the lower resolution sample points E-H to cache tag block 1015" and "[i]f a miss occurs, then <u>cache tag blocks 1010 and 1015 forward</u> a cache write address to memory request generator 1020" which "<u>generates memory requests for all misses</u>… <u>and forwards the requests to DRAM 655 for information retrieval</u>" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). Further, Rivard discloses that "DRAM

655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added).

However, the disclosure of a memory request generator that generates memory requests for all misses and forwards the requests to DRAM for information retrieval, and that the DRAM then returns the memory data which is stored at the write address, as in Rivard, simply fails to even suggest "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as claimed by applicant. Clearly, returning memory data which is stored at a write address after a cache miss, as in Rivard, fails to meet "receiving instructions from the video memory in response to the instruction request" (emphasis added), as claimed by applicant. Applicant emphasizes that the memory data from the DRAM, as disclosed in Rivard, fails to teach or suggest "instructions," in the manner as claimed by applicant.

In addition, the Examiner has admitted that "Rivard does not explicitly teach to combine texture mapping stage and texel cache system to form a texture module," and has argued that "it would have been obvious to one of ordinary skill in the art at the time of present invention to combine texture mapping stage and texel cache system of Rivard to work together as a texture module."

Applicant disagrees and respectfully asserts that it would not have been obvious to combine the texture mapping stage and texel cache system in Rivard, as suggested by the Examiner. Specifically, Rivard teaches the use of "pipeline latency elements 1025" which "coordinate arrival of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (Col. 7, lines 3-7 - emphasis added). However, Rivard actually *teaches away* from the Examiner's above allegation, in addition to applicant's claim language, by intentionally incorporating the pipeline latency elements to coordinate arrival of the memory data from separate sources. Therefore, for the reasons argued above, it would not be obvious to "combine texture mapping stage and texel cache system," as argued by the Examiner. Thus, it would not have been obvious to "receiv[e] instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as applicant claims.

Applicant thus formally requests a specific showing of the subject matter in ALL of the claims in any future action. Note excerpt from MPEP below.

> "If the applicant traverses such an [Official Notice] assertion the examiner should cite a reference in support of his or her position." See MPEP 2144.03.

Still yet, the Examiner has admitted that "Rivard does not explicitly teach that the memory returns instructions along with data, in response to [the] instruction request from the texture module," but has argued that "Wang teaches a graphics subunit (texture module) in a graphics hardware system that supplies data and control signals to local frame buffer memory for executing a series of display instructions (Fig. 1, col. 5 lines 38-67…)."

Applicant respectfully disagrees. The excerpt from Wang relied on by the Examiner merely discloses a "graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory." However, only generally disclosing that a graphics subunit executes instructions stored in computer memory, as in Wang, fails to specifically meet applicant's claimed "receiving instructions from the video memory in response to the instruction request utilizing the texture module in the graphics pipeline" (emphasis added), as applicant claims. In fact, applicant notes that Wang only discloses that the "graphics subunit 109 includ[es] a texture engine 10, [and] a polygon engine 12," and that the "texture engine 10 is responsible for retrieving the texture map data," whereas the "polygon engine 12…performs well known polygon rendering functions" (Col. 6, lines 3-49). Thus, Wang clearly does not disclose "receiving instructions…utilizing the texture module in the graphics pipeline" (emphasis added), as claimed.

Moreover, with respect to independent Claim 28, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 in Rivard to make a prior art showing of applicant's claimed "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory."

Applicant again respectfully asserts that figures from Rivard relied upon by the Examiner only shows a block diagram of a computer system, and that the excerpts relied on by the Examiner simply disclose that "the cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" where "[m]emory request generator 1020 generates memory requests for all misses." Clearly, generating a <u>memory request</u> for all misses, as in Rivard, does not meet applicant's claimed "sending an <u>instruction request</u>" (emphasis added), as claimed. Applicant also respectfully points out the arguments made above with respect to at least some of the other independent claims which clearly show that Rivard does not meet applicant's specific claim language.

In the Office Action mailed 03/20/2007, the Examiner has argued that "Rivard teaches this limitation" and to "refer to the rejection of claim 1… regarding sending an instruction request to video memory, where a texture module in a graphics pipeline sends the instruction request to the video memory." Applicant disagrees with the Examiner's arguments and asserts that, for substantially the same reasons as those argued above with respect to at least some of the independent claims, Rivard fails to even suggest "sending an instruction request," as applicant claims. For example, applicant respectfully points out that Rivard merely teaches that "the <u>memory request</u> generator is coupled between the cache tag block and the cache data store for performing a <u>memory request before</u> an address and instruction information associated with the <u>needed texels</u> reach the cache data store" (Col. 10, lines 48-52 - emphasis added). Clearly, performing a <u>memory request before</u> the address and instruction information reach the cache data store, as in Rivard (see excerpts above), fails to teach "sending an <u>instruction request</u>" (emphasis added), as claimed by applicant.

Furthermore, with respect to Claim 28 and the Examiner's reliance upon the rejection of Claim 1, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Applicant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Applicant respectfully asserts that Rivard merely discloses that the "cache tag blocks 1010 and 1015 forward a cache write address to memory request generator 1020" which "generates memory requests for all misses... and forwards the requests to DRAM 655 for information retrieval" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM for information retrieval, as in Rivard, simply fails to suggest "sending an instruction request to video memory, where the texture module sends the instruction request to the video memory" (emphasis added), as claimed by applicant. Clearly, generating and forwarding a memory request to DRAM, as in Rivard, fails to suggest "sending an instruction request" (emphasis added), in the manner as claimed by applicant.

In addition, with respect to Claim 28, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard to make a prior art showing of applicant's claimed "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module."

Specifically, the Examiner has argued that "Rivard... teaches that DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage." Further, the Examiner has argued that "the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module." In addition, the Examiner has argued that "[t]he texture mapping stage as shown in Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions."

Applicant respectfully disagrees with the Examiner's arguments and asserts that Rivard merely discloses that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added). However, merely storing memory data at the write address, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by applicant. Clearly, storing memory data, as in Rivard, fails to suggest "receiving additional instructions," in the manner as claimed by applicant.

Second, in response to the Examiner's argument that "the texture mapping stage and the texel cache system together are considered as texture module," applicant respectfully disagrees and asserts that Rivard merely teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage is separate from the texture cache system, as in Rivard, which fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by applicant.

Third, in response to the Examiner's argument that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions," applicant respectfully disagrees and asserts that Rivard merely teaches that "the <u>memory request generator</u> is coupled between the cache tag block and the cache data store for performing <u>a memory request</u> <u>before</u> an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "<u>memory data</u>", where "the data is conveniently aligned in the <u>texture map</u>" (emphasis added), as in Rivard (see excerpts above), fails to teach "receiving <u>additional instructions</u>" (emphasis added), as claimed by applicant. In addition, applicant respectfully asserts that simply nowhere in the description of Figure 6 is there any disclosure that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline," as noted by the Examiner. Thus, Rivard simply fails to meet applicant's claimed "receiving additional instructions... utilizing the texture module," as claimed.

Furthermore, with respect to independent Claim 29, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard, along with Fig. 3; and Page 5, lines 24-31 from AAPA to make a prior art showing of applicant's claimed "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory."

Applicant respectfully asserts that the figures relied on by the Examiner only generally illustrate a computer system and a block diagram detailing graphics accelerator system showing the pipeline latency elements 1025. In addition, in Col. 7, lines 4-7, Rivard teaches that "graphics accelerator system 635 <u>includes pipeline latency elements 1025</u> to <u>coordinate arrival</u> of the memory data and of the associated instructions at cache data store and memory data resolver 1030" (emphasis added). Furthermore, Rivard teaches that "[m]emory request generator 1020 <u>generates memory requests for all misses</u>

(up to four for bi-linear sampling and up to eight for tri-linear sampling), and forwards the requests…" (Col. 6, lines 50 – 52 emphasis added). However, disclosing that memory requests are generated for all misses, as in Rivard, fails to suggest "sending an instruction request to memory utilizing a texture module coupled to the shader module" (emphasis added), as claimed by applicant.

In addition, applicant respectfully asserts that Fig. 3 from AAPA as relied upon by the Examiner merely discloses that "the texels resulting from one texture lookup can influence the location of the texels in a subsequent texture lookup" (Page 4, lines 30-31 – emphasis added). However, the mere disclosure of texels resulting from a texture lookup fail to even suggest "sending an instruction request to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by applicant. Clearly, retrieving texels from a texture lookup fails to meet "an instruction request," in the manner as claimed by applicant.

In the Office Action mailed 03/20/2007, the Examiner has argued that "Rivard teaches sending an instruction request to video memory, where the texture module sends the instruction to the video memory." Applicant disagrees with the Examiner's arguments and asserts that, for substantially the same reasons as those argued above with respect to at least some of the independent claims, Rivard fails to even suggest "sending an instruction request," as applicant claims. For example, applicant respectfully points out that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (Col. 10, lines 48-52 - emphasis added). Clearly, performing a memory request before the address and instruction information reach the cache data store, as in Rivard (see excerpts above), fails to teach "sending an instruction request" (emphasis added), as claimed by applicant.

Furthermore, with respect to Claim 29 and the Examiner's reliance upon the rejection of Claim 1, the Examiner has argued that "the definition of instruction provided by dictionary.com, which defines instructions as a command given to a computer to carry out a particular operation and can contain data to be used in the operation; here DRAM sends memory data based on the memory requests/read requests from the texture module." Further, the Examiner has argued that "[m]emory requests/read requests act as an instruction to DRAM, which performs a particular function based on the request."

Applicant respectfully disagrees with the Examiner's arguments and asserts that the dictionary.com reference provided by the Examiner merely defines an instruction as "a command given to a computer to carry out a particular operation." Applicant respectfully asserts that Rivard merely discloses that the "<u>cache tag blocks 1010 and 1015 forward</u> a cache write address to <u>memory request generator</u> 1020" which "<u>generates memory requests for all misses</u>... <u>and forwards the requests to DRAM 655 for information retrieval</u>" (Col. 6, lines 33-36, and lines 48-53 – emphasis added). However, the disclosure of the memory request generator generating a memory request and forwarding the request to DRAM <u>for information retrieval</u>, as in Rivard, simply fails to specifically suggest "sending <u>an instruction request</u> to video memory, where a texture module coupled to the shader module sends the instruction request to the video memory" (emphasis added), as claimed by applicant. Clearly, generating and forwarding <u>a memory request</u> to DRAM, as in Rivard, fails to suggest "sending <u>an instruction request</u>" (emphasis added), in the manner as claimed by applicant.

In addition, with respect to Claim 29, the Examiner has relied on Figures 6 and 10; Col. 4, lines 46-57; Col. 6, lines 45-67; and Col. 7, lines 1-10 from Rivard, along with Fig. 3; and Page 5, lines 24-31 from AAPA to make a prior art showing of applicant's claimed "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module."

Specifically, the Examiner has argued that "Rivard... teaches that DRAM returns memory data to cache data store and memory data resolver component of texel cache system, which further sends this information to the texture mapping stage." Further, the Examiner has argued that "the texture mapping stage and the texel cache system together are considered as texture module, so the information is passed between the components of the texture module." In addition, the Examiner has argued that "[t]he texture mapping stage as shown in Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions."

Applicant respectfully disagrees with the Examiner's arguments and asserts that Rivard merely discloses that "DRAM 655 returns the memory data on bus 660 to cache data store and memory data resolver 1030, which stores the memory data at the write address" (Col. 6, lines 53-56 – emphasis added). However, merely storing memory data at the write address, as in Rivard, simply fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by applicant. Clearly, storing memory data, as in Rivard, fails to suggest "receiving additional instructions," in the manner as claimed by applicant.

Second, in response to the Examiner's argument that "the texture mapping stage and the texel cache system together are considered as texture module," applicant respectfully disagrees and asserts that Rivard merely teaches that "[g]raphics accelerator system 635 includes graphics pipeline stages 640 including a texture mapping stage 645 for mapping texture information to the graphics information received from graphics application program 670 and for maintaining the texel information in a texel cache system 650" where the "[t]exture mapping block 645 sends information via bus 647 to texel cache system 650, and texel cache system 650 sends information via bus 649 back to texture mapping block 645" (Col. 4, lines 49-57 – emphasis added). Clearly, the texture mapping stage and separate texture cache system, as in Rivard, fails to suggest "receiving additional instructions from the video memory in response to the instruction request utilizing the texture module" (emphasis added), as claimed by applicant.

Third, in response to the Examiner's argument that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline, and thus the instructions received from the DRAM via the texel cache system are considered to be the additional instructions," applicant respectfully disagrees and asserts that Rivard merely teaches that "the memory request generator is coupled between the cache tag block and the cache data store for performing a memory request before an address and instruction information associated with the needed texels reach the cache data store" (emphasis added). Clearly, coordinating the arrival of "memory data", where "the data is conveniently aligned in the texture map" (emphasis added), as in Rivard (see excerpts above), fails to teach "receiving additional instructions" (emphasis added), as claimed by applicant. In addition, applicant respectfully asserts that simply nowhere in the description of Figure 6 is there any disclosure that "Fig. 6 also receives data (instructions) from the rasterizer module of the pipeline," as noted by the Examiner. Thus, Rivard simply fails to meet applicant's claimed "receiving additional instructions… utilizing the texture module," as claimed.

Applicant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, since it would be *unobvious* to modify the Rivard reference, as noted above, and the prior art references, as relied upon by the Examiner, fail to teach or suggest all of the claim limitations, as noted above. Thus, a notice of allowance or a proper prior art showing of all of applicant's claim limitations, in combination with the remaining claim elements, is respectfully requested.

Applicant further notes that the prior art is also deficient with respect to the dependent claims. For example, with respect to Claim 18, the Examiner has relied on Col. 5, lines 43-67; and Col. 6, lines 1-47 from Wang to make a prior art showing of applicant's claimed technique "wherein a complete instruction set is received in response to the instruction request." Specifically, the Examiner has argued that "Wang… teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory" and that "[t]he graphics subunit receives display instructions including texture data based on the supplied data and control signals."

Applicant respectfully disagrees and asserts that Wang merely discloses that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory," where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). However, only generally disclosing the graphics subunit executing a series of display instructions found within a display list stored in computer memory, as in Wang, does not specifically teach that "a complete instruction set is received in response to the instruction request," where the "instructions [are received] from the video memory" (emphasis added), in the context as claimed by applicant (see Claim 1 for context).

In addition, with respect to Claim 19, the Examiner has relied on Col. 5, lines 43-67; and Col. 6, lines 1-47 from Wang to make a prior art showing of applicant's claimed technique "wherein a partial instruction set is received in response to the instruction request." Specifically, the Examiner has argued that "Wang… teaches a graphics subunit (texture module) of a graphics hardware system executing a series of display instructions (set of instructions) stored in computer memory" and that "[t]he graphics subunit receives display instructions including texture data based on the supplied data and control signals."

Applicant respectfully disagrees and asserts that Wang merely discloses that "[t]he graphics hardware system 108 contains a 3D graphics subunit 109 for executing a series of display instructions found within a display list stored in computer memory" where "[t]he display list generally contains instructions regarding the rendering of several graphic primitives, e.g., individual points, lines, polygons, fills, BIT BLTs (bit block transfers), textures, etc." (Col. 5, lines 40-46 – emphasis added). However, only generally disclosing the graphics subunit executing a series of display instructions found within a display list stored in computer memory, as in Wang, does not specifically teach that "a partial instruction set is received in response to the instruction request," where the "instructions [are received] <u>from the video memory</u>" (emphasis added), in the context as claimed by applicant (see Claim 1 for context).

In fact, applicant notes that the Examiner has relied on the same disclosure in Wang of a "series of display instructions" to meet both applicant's claimed "complete instruction set [that] is received in response to the instruction request" (see Claim 18) and "partial instruction set [that] is received in response to the instruction request" (see Claim 19), as claimed. Clearly, a generally disclosure of a series of instructions, as in Wang, simply cannot meet applicant's claimed "complete instruction set" (Claim 18) <u>and</u> "partial instruction set" (Claim 19), as claimed.

Again, applicant respectfully asserts that at least the first and third elements of the *prima facie* case of obviousness have not been met, as noted above. Thus, a notice of allowance or specific prior art showing of each of the foregoing claim elements, in combination with the remaining claimed features, is respectfully requested.

To this end, all of the independent claims are deemed allowable. Moreover, the remaining dependent claims are further deemed allowable, in view of their dependence on such independent claims.

In the event a telephone conversation would expedite the prosecution of this application, the Examiner may reach the undersigned at (408) 505-5100. The Commissioner is authorized to charge any additional fees or credit any overpayment to Deposit Account No. 50-1351 (Order No. NVIDP064).

Respectfully submitted,
Zilka-Kotab, PC.

/KEVINZILKA/

Kevin J. Zilka
Registration No. 41,429

P.O. Box 721120
San Jose, CA 95172-1120
408-505-5100